Army Research Laboratory

Framework for a Software-defined Global Positioning System (GPS) Receiver for Precision Munitions Applications

by Mark Ilg

ARL-TR-5976 April 2012

NOTICES

Disclaimers

The findings in this report are not to be construed as an official Department of the Army position unless so designated by other authorized documents.

Citation of manufacturer's or trade names does not constitute an official endorsement or approval of the use thereof.

Destroy this report when it is no longer needed. Do not return it to the originator.

Army Research Laboratory
Aberdeen Proving Ground, MD 21005

ARL-TR-5976 **April 2012**

Framework for a Software-defined Global Positioning System (GPS) Receiver for Precision Munitions Applications

Mark Ilg
Weapons and Materials Research Directorate, ARL

REPORT DOCUMENTATION PAGE	***Form Approved*** ***OMB No. 0704-0188***

1. REPORT DATE *(DD-MM-YYYY)* April 2012	**2. REPORT TYPE** Final	**3. DATES COVERED (From - To)** June 2010 to July 2011

4. TITLE AND SUBTITLE

Framework for a Software-defined Global Positioning System (GPS) Receiver for Precision Munitions Applications

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

Mark Ilg

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

U.S. Army Research Laboratory
ATTN: RDRL-WML-F
Aberdeen Proving Ground MD 21005

8. PERFORMING ORGANIZATION REPORT NUMBER

ARL-TR-5976

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release; distribution unlimited.

13. SUPPLEMENTARY NOTES

14. ABSTRACT

Gun-launched munitions rely heavily on the global positioning system (GPS) receivers for position, velocity, and "up-finding" in their guidance, navigation, and control (GN&C) systems. These receivers account for substantial cost in the GN&C electronics due to the contractor intellectual property and non-recurring engineering (NRE) costs. Typically, the designs are not optimized for gun-launched munitions, but rather as general purpose receivers for missiles, unmanned aerial vehicles, aircraft, man-portable location systems, and vehicles. Due to the high launch dynamics, short time of flight, high spin rate, etc., the general purpose receivers have trouble acquiring and locking on the weak GPS signals. To address these challenges, software algorithms tailored to the flight dynamics can assist in the basic features of the receiver as well as provide jamming and spoofing immunity. In this report, a software-defined radio (SDR) approach is used for the development of global navigation satellite systems (GNSS) receivers. This approach provides an evaluation capability for algorithm development and evaluation that is nearly hardware agnostic.

15. SUBJECT TERMS

GPS, SDR

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON** Mark Ilg
a. REPORT	**b. ABSTRACT**	**c. THIS PAGE**			**19b. TELEPHONE NUMBER** *(Include area code)*
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UU	34	(410) 306-0780

Standard Form 298 (Rev. 8/98)
Prescribed by ANSI Std. Z39.18

Contents

List of Figures

List of Tables

1. Introduction

Software-defined radio (SDRs) is a growing field due to the higher throughput of general purpose processors and digital signal processors (DSP). On the receiver portion of the SDR the radio frequency (RF) signal is down-converted to a baseband digital signal instead of multiple intermediate frequencies (IF) via a high-speed analog to digital converter (ADC). This signal is then processed in the digital domain to extract the data. Analog would still play a part in the system design, limiting noise bandwidth, anti-aliasing, and minimizing external interference; however, the bulk of the signal processing would be performed digitally. One example of military use of SDR is the Joint Tactical Radio System (JTRS), which is envisioned as the backbone of military communications and is intended to link manned and unmanned systems *(3)*. Although this technology has been adopted by the military, the technology is not used in global positioning system (GPS) receivers.

Currently, there are three gun-launched munitions in the field that used GPS receivers, all of which use GPS receivers. These receivers are highly capable and include advanced methods for direct P(Y) acquisition, tracking, and navigation. These receivers are not designed specifically for use in gun-launched munition environments and are ”tuned” for general purpose GPS requirements. Due to the high launch dynamics, short time of flight, high spin rate, etc., general purpose receivers have trouble acquiring and locking on weak GPS signals. To address these challenges, software algorithms tailored to the flight dynamics can assist in the basic features of the receiver as well as provide jamming and spoofing immunity. Currently, the Department of Defense (DoD) has no capability within the Government to provide solutions to these problems and the commercial GPS industry has an extremely limited capability. To assist in the development of solutions to these problems, and to incorporate the latest GPS signalling (multi-frequency and M-Code), a SDR approach is taken for algorithm development tailored to the gun-launched munition environment. An outline of the end use of the system described in this report is shown in figure 1. The objective of this project is to use the SDR frontend to capture GPS data for use in Precision Simulation Environment (PRESIMEN). The SDR will provide “real-world” input data for validating GPS software models, data sets for analyzing multi-domain algorithms, and integration into hardware-in-the-loop (HIL) experiments for real-time verification of algorithms.

This report outlines the development of the software architecture for capturing GPS signals for future algorithm development. Reliable signal acquisition of the GPS waveform is challenging due to the high sample rates and bandwidth required for the spreading code of the GPS signal. This report contains of a brief description of the GPS signal in section 2 while section 3 describes the commercial-off-the-shelf (COTS) hardware used in the design for the sampling of the RF data. Section 4 details the communication between the

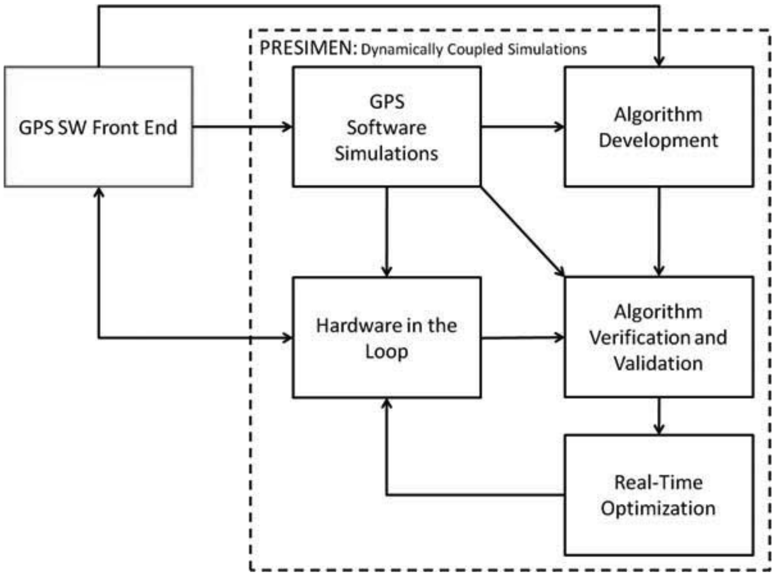


Figure 1. PRESIMEN Integration.

hardware and a personal computer (PC) used data recording. The software components consisting of the device firmware, windows drivers, graphical user interface (GUI), and post-processing are described in section 5. Finally, results of implementation of a SDR GPS receiver are presented in section 6, followed by concluding remarks in section 7.

2. GPS Signal

Although the focus of this report is on the software acquisition of GPS RF data, it is important to understand the fundamental structure of the GPS signal. The purpose of this section is to show how the GPS signal is converted from the raw RF, data as generated by the satellite, and then is transformed into a inphase and quadrature (IQ) data stream for processing. The GPS signal is actually a code division multiple access (CDMA) waveform consisting of two spreading codes, clear/acquisition (C/A) and the encrypted precision P(Y) code. In this report we limit our discussion to C/A codes only since processing of the P(Y) code is similar and the hardware becomes more complicated. This spreading converts a lower bandwidth signal, the GPS navigation message, to a higher bandwidth signal with the ability to co-host signals on the same carrier frequency with the added benefit of in-band interference suppression. Equation 1 is the GPS signal at the L1 carrier frequency of 1575.42 MHz. The C/A spreading code (C^n), or Gold code, is a psuedorandom signal binary sequence generated by shift registers with statistical properties similar to noise *(6)*. Each Space Vehicle NAVSTAR (SVN) has a unique psuedorandom noise (PN) sequence, which are all nearly orthogonal. The signals are modulated using binary phase shift key (BPSK) onto the carrier signal, $\cos(2\pi f_{L1}t)$. Table 1 lists the GPS signal parameters.

$$s_{L1}^{n}\left(t\right)=\sqrt{2P_{C/A}}C^{n}\left(t\right)D^{n}\left(t\right)\cos(2\pi f_{L1}t)+\sqrt{2P_{P(Y)}}P^{n}\left(t\right)D^{n}\left(t\right)\sin(2\pi f_{L1}t) \tag{1}$$

Table 1. GPS signal parameters.

	Description	Typical Value	Units
$P_{C/A}$	C/A Signal Power	26.8	dBW
$P_{P(Y)}$	P(Y) Signal Power	$P_{C/A}-3$ dB	dBW
$C^{n}\left(t\right)$	C/A Spreading Code	1.023	Mbps
$P^{n}\left(t\right)$	P(Y) Spreading Code	10.23	Mbps
$D^{n}\left(t\right)$	Navigation Message	50	bps
f_{L1}	L1 Center Frequency	1575.42	MHz

The minimum power of 26.8 dBW (including GPS satellite antenna gain) of the generated C/A signal is broadcast from the satellite through the atmosphere to the receiver. This loss can be approximated to be the sum of the free space path loss, approximately 182.4 dB, atmospheric loss, 2–3 dB, and receiver antenna gain, +1 dB to <-40 dB *(9)*, which will vary with projectile geometry *(11)*. Therefore, fundamentally, in the optimal scenario, the C/A code at the receiver is approximately -157.4 dBW (-127.4 dBm).

$$P_{N}=k_{B}Ts_{BW} \tag{2}$$

The noise floor of a receiver is mostly due to the thermal noise, or Johnson-Nyquist noise, given by equation 2, where P_N is the thermal noise power, k_B is Boltzman constant, T is the temperature in kelvin, and s_{BW} is the signal bandwidth. For the GPS bandwidth for a C/A code receiver of approximately 2 MHz, a room temperature of 290 K, the noise floor is approximately -110.86 dBm (figure 2). To recover the navigation message, $D^{n}\left(t\right)$, communication signal processing techniques are used to pull the GPS signal above the noise floor. For more information on the signalling structure of GPS, refer to references *(10, 1, 5, 7, 8, 12)*.

At the receiver, the signal is down converted to an IF by means of RF circuitry. One of the primary steps of the GPS receiver RF circuitry is to amplify the low signal high enough for sampling via an ADC, typically on the order of 1–2 V pp, or 4–10 dBm, for a mid-range ADC *(2)*. When amplifying a low-level signal such as the GPS signal, one has to ensure that the noise figure (NF) of each of the components does not diminish the overall carrier-to-noise (C/N_o) of the recovered signal. Each component in the RF chain contributes to the overall NF, F_T, via the following equation *(4)*, where F_n and G_n are the NF and gain of the n^{th} component in the chain, respectively.

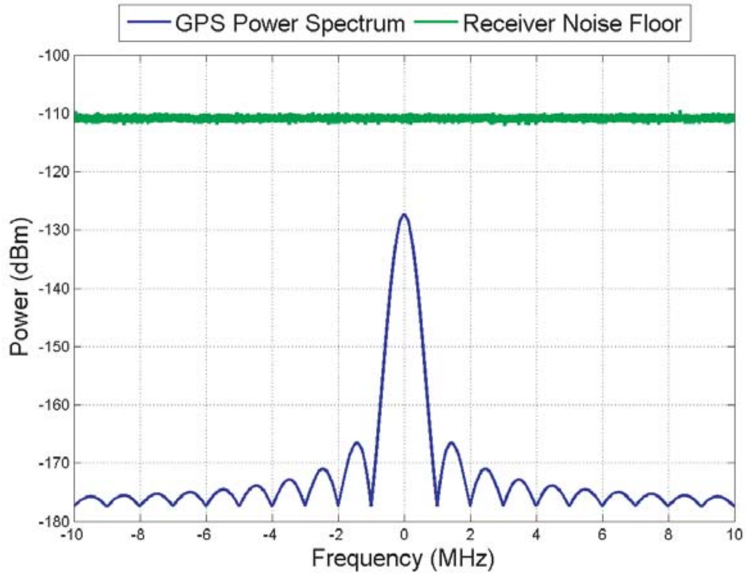


Figure 2. GPS signal.

$$F_T = F_1 + \frac{F_2 - 1}{G_1} + \cdots + \frac{F_N - 1}{\prod_{n=1}^{N-1} G_n} \qquad (3)$$

As an example, figures 3 and 4 show two possible configurations for a GPS frontend amplification stage, from an input port, a, to an output port, b. Where the components are identical, the NF for figure 3 is 0.7931 dB and the NF for figure 4 is 2.7560 dB, a substantial decrease in signal level. Therefore, we can minimize the noise contribution due to the components by placing a low-noise amplifier (LNA) at the antenna feed, known as an active antenna, to minimize the loss of the cable to the receiver's first amplification stage.

Figure 5 depicts the typical RF frontend for a GPS receiver with an active antenna.

The signal is first amplified at the antenna via the LNA and passed to the receiver over low-loss cabling. The bias-T is an RF component that passes DC along with the RF to provide power to the LNA. This stage is followed by another LNA, a band-pass filter

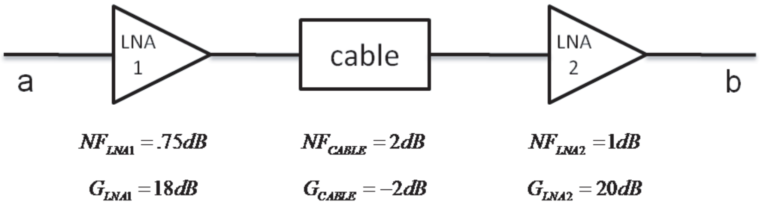


Figure 3. Noise figure (a).

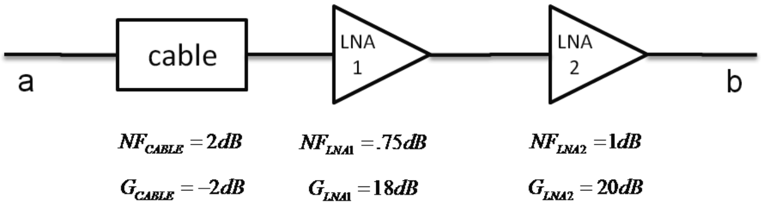


Figure 4. Noise figure (b).

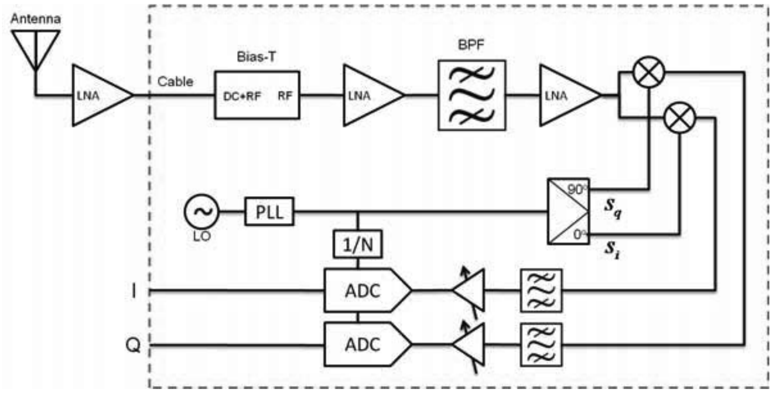


Figure 5. RF frontend.

centered around the L1 carrier frequency, and finally another LNA. The next step is to mix the incoming RF with locally generated carrier signals, s_i and s_q, the in-phase and quadrature to the incoming signal, respectively. This mixed signal, is then low-pass filtered to recover the signal at an IF, which is substantially less than the carrier frequency f_c and suitable for sampling with an ADC. The final stage prior to sampling is a variable-gain amplifier (VGA), typically controlled by an automatic gain control (AGC). The AGC is used to provide a larger dynamic range for the ADC as well as maintain a good probability distribution function (DF) on the incoming signal. At this point, the IQ data is in a usable format for stream processing and algorithm development.

3. Hardware Components

To facilitate the design of the SDR, a universal serial bus (USB) COTS GPS frontend, the silicon germanium (SiGe) GN3S Sampler v2, figure 6, was chosen as a base configuration and later expanded to include the a Maxim 2769 GPS receiver evaluation kit. Again, for ease of design and limitations on the sampling rate, only C/A signals are considered in this application as the C/A code algorithms should port to the P(Y) code. The SiGe GN3S Sampler v2 consists of two primary integrated circuit (IC) chips, the SE4120L and CY7C68013A.

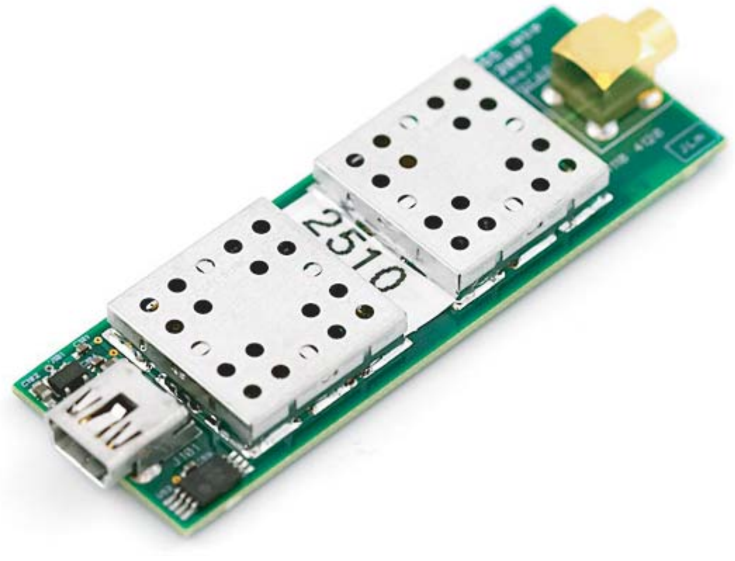


Figure 6. GN3S Sampler v2.

The SE4120L is a complete GPS frontend IC manufactured by SiGe Semiconductor, figure 7, with a 2.2 dB typical RF NF. The SE4120L contains a 2-bit dual ADC, for IQ signals, with selectable sample rates, data output format, and serial output format, as summarized in table 2.

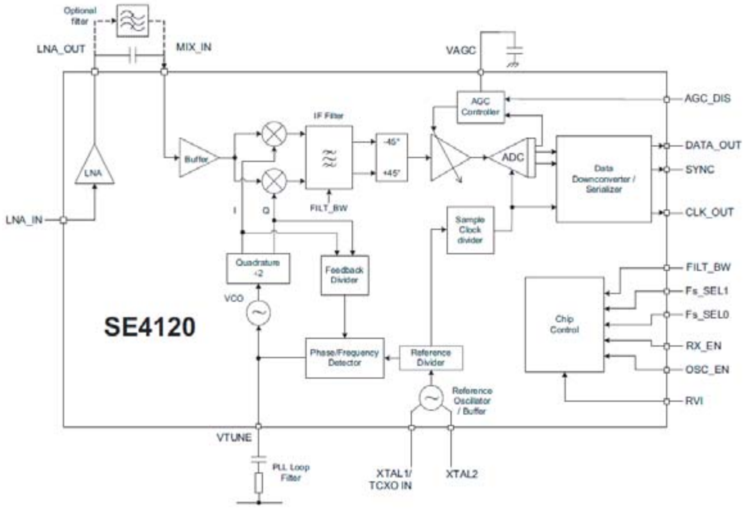


Figure 7. SE4120L *(17)*.

The IQ ADC data is available via a three-wire interface containing the DATA_OUT signal, SYNC signal, and CLK_OUT signal. The IC configuration, consisting of the control of the onboard AGC circuitry, which provides a dynamic range of 50 dB, and the bandwidth of the IF filter, selectable between 2.2 and 4.4 MHz, is achieved through a series of logic pins.

Table 2. SE4120L ADC output.

Sampling Rate	Data Output Format	Serial Output Format
8.184MSPS	2-bit I/Q	Pulse
5.456MSPS	2-bit I/Q	Byte
4.092MSPS	2-bit I/Q	Byte
4.092MSPS	4-bit I/Q	Pulse

The reference oscillator for the system is chosen to be a temperature compensated crystal oscillator (TCXO) with a nominal center frequency of 16.368 MHz, providing a near 0 Hz IF.

The CY7C68013A, figure 8, is a USB 2.0 IC manufactured by Cypress Semiconductor. The CY7C68013A consists of a 8051 micro-controller, high-speed USB transceiver, programmable interfaces, onboard RAM, integrated 16 bit wide first in, first out (FIFO) capable of 96 MBytes/s burst data rates. The onboard RAM is loaded via an erasable programmable read-only memory (EEPROM) on the GN3S Sampler or via USB. The USB is capable of data rates of 480 Mbps, for high-speed communication to a PC *(13)*.

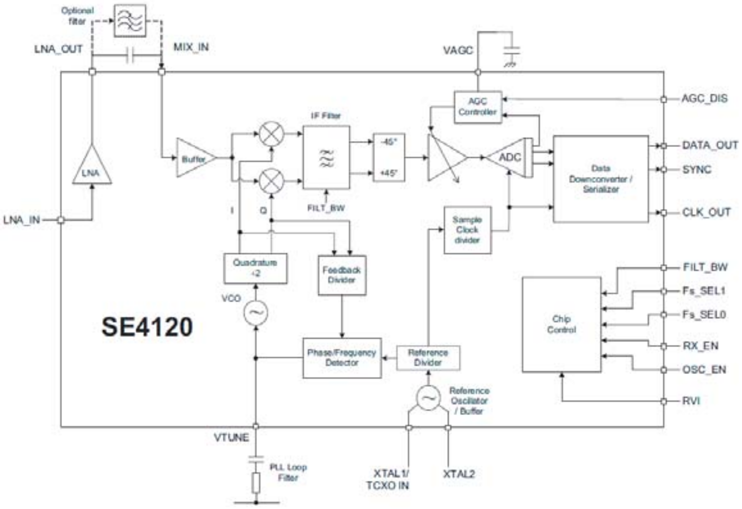


Figure 8. CY7C68013A *(13)*.

Connections are made between CY7C68013A and the SE4120L to include the three-wire data interface (DATA_OUT, SYNC, CLK_OUT) control of the IF filter bandwidth and data format. Table 3 summarizes the physical connections and pertinent signals.

The AGC is forced on by a pull-down resistor. The AGC operates by a closed-loop control on the pre-ADC gain though logic based on the magnitude of the complex IQ data, ensuring the magnitude is high approximately 33% of the time. This AGC loop time constant is prescribed by an external capacitor, yielding a value of 10 ms.

The CY7C68013A communicates through the USB 2.0 standard interface to the PC. This communication interface is detailed in section 4. The physical hardware connections (figure 9) are prescribed by the USB standard as a four-wire interface containing a differential signal, D+ and D-, power, and ground.

Table 3. IC connections.

SE4120L	CY7C68013A	Description
DATA	FD[0]	IQ Data
CLK	IFCLK	IQ Clock
SYNC	FD[1]	Synchronization of IQ Data
Fs_SEL0	FD[12] (PD4)	Stream Configuration
Fs_SEL1	FD[11] (PD3)	Stream Configuration
Filt_BW	FD[10] (PD2)	IF Filter Bandwidth
RX_EN	Pull-Up	Receiver Enable (Always On)
OSC_EN	Pull-Down	Using TCXO
AGC_DIS	Pull-Down	AGC Enabled

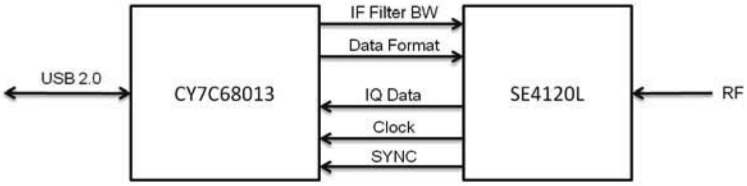


Figure 9. Hardware connections.

4. Communications

Two separate communications channels exist in the SDR platform, the SE4120L to CY7C68013A communications and the CY7C68013A to PC USB communications, each of which is described in this section.

4.1 SE4120L to CY7C68013A Communications

As described by the physical connections in section 3, the SE4120L communicates to the CY7C68013A over the FIFO parallel interface, known as the general programmable interface (GPIF) (figure 10). The GPIF serves as a “glue logic” between external components and the internal FIFO memory of the device. This provides hardware arbitration between an external device and the USB portion of the CY7C68013A, minimizing the processor burden and increasing the throughput of external devices to the USB interface. Control of the GPIF interface is governed by a state machine architecture set up by the 8015 micro-controller through software registers. These software registers within the device provide information regarding waveforms supplied to the GPIF physical pins of the CY7C68013A. Programming of these registers is hard-coded into the device via firmware and the use of a Windows-based GPIF designer tool, GPIFtool.exe *(11)*.

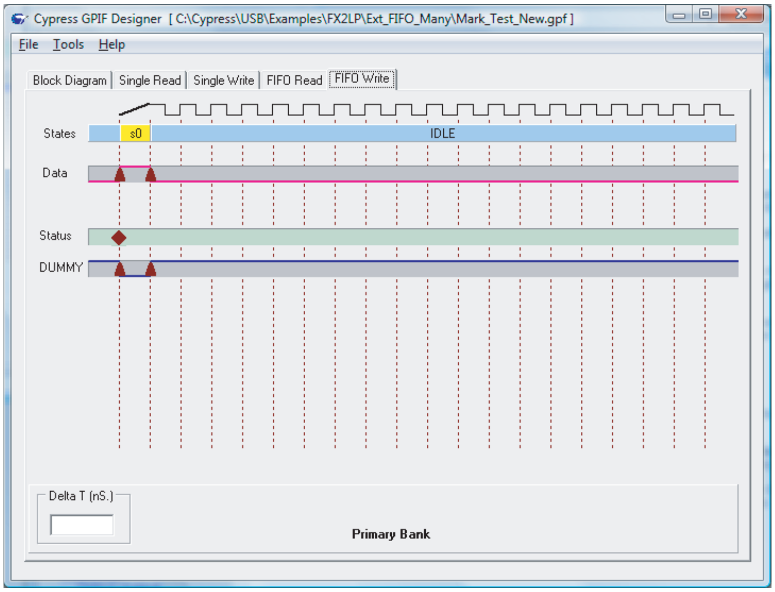


Figure 10. GPIF tool.

Since only two data bits are required for the SE4120L’s data stream, the IQ serial data (FD[0]) and Sync (FD[1]), the GPIF is configured for 8-bit wide parallel access. This frees up the upper 8-bits for the control signalling. In this configuration, the SE4120L controls the clocking signals for the GPIF to insure proper sampling of the GPS IF data. In this instance, the GPIF samples the FD bus on the rising edge of the IF clock (IFCLK) and transfers the data in a serial data stream to the FIFO.

The FIFO has shared memory for an USB endpoint, which after initialization, allows for direct pass-through of data from the device to the USB interface. Once initialized, data from the GPIF pins (FD) will continuously fill the FIFO; only on overflow will the 8015 arbitrate though a serviceable hardware interrupt. Since, we only require one-way communication on the FD lower bits, the GPIF is only configured for FIFO burst write and is stuck in an infinite loop. The interaction between the GPIF and the FIFO is shown in figure 11. For more detailed information regarding the functionality of the GPIF and FIFO, see reference 15 and 14.

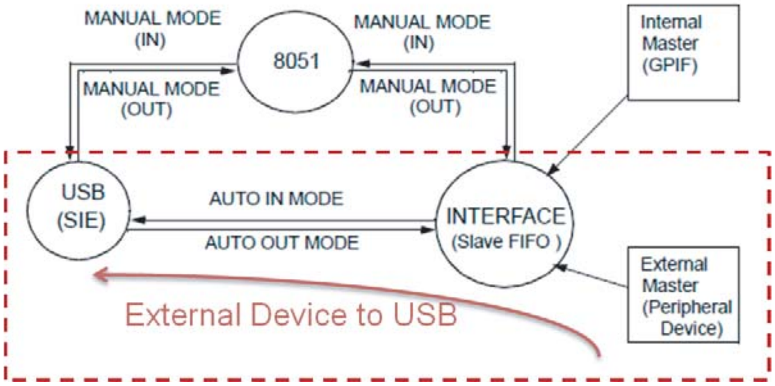


Figure 11. GPIF FIFO interaction.

4.2 CY7C68013A to PC Communications

The CY7C68013A communicates via USB to the host computer. The USB consists of layers of protocols, where the lower levels are handled though on-chip hardware abstraction. The USB 2.0 standard, which can be downloaded as of publication via www.usb.org, contains all the definitions of the physical signalling of the USB signalling as well as the protocols. For this report, we focus on the type of communication used in this application and how it is applied.

The method for communicating the data from the aforementioned FIFO is via “endpoints.” Endpoints are sources or sinks of data to/from a device, handling the interactions between the firmware and the hardware. In this application, we will use two endpoints, EP0 and EP6, for handling transactions between the PC and the device. Endpoint Zero, EP0, is an endpoint required by all USB-compliant devices and provides the link for device control and status requests, most importantly during the enumeration process. Endpoint 6, EP6, is an endpoint that can be configured for high-speed data transfer in 512-byte packets. In streaming applications (video camera applications, audio devices, etc.), two types of endpoints are available, isochronous and bulk endpoints. Isochronous endpoints have a guaranteed bandwidth but not guaranteed delivery; therefore, since the bulk endpoint provides more reliable data, we use that transfer protocol. As long as the FIFO does not overflow and the PC can service the USB data integrity should not be compromised, is addressed in section 5.

5. Software

The software for this application can be divided into several categories: device firmware, USB windows driver, capture software, and validation software. The device firmware is the software that resides on the CY7C68013A IC. The USB windows driver is a mechanism for handling the hardware-software interface within the Microsoft Windows environment. The capture software is a GUI for taking information from the driver and interacting with the user. Finally, the validation software is a package of GPS routines to validate the captured IQ data so it can properly decode the GPS information, including position, velocity, and time, for future research.

5.1 Device Firmware

The device firmware consists of several elements built around the EZ-USB Firmware Framework developed by Cypress Semiconductor, figure 12. This method was chosen for ease of development and quick execution of the software design. On startup, the device loads the application from an external EEPROM. This application gets information about the USB configuration through the device descriptor, including data such as the product identification (PID), vendor identification (VID), and endpoint information. The PID and VID are regulated by USB.org; however, since this is not a commercial product, the Cypress default development VID was chosen along with a random VID.

The firmware framework handles all of the USB-specific startup and messaging and points to pre-defined user function calls within the user firmware. For example, the device receives a message on EP0, the framework handles the input request and passes the functional parameters to a user defined-function handler. This removes the user software requirement of handling all of the information contained within the data packets to/from the PC. The user segment of the code handles all device specific requests and also the configuration of the GPIF interface. The following list is a summary of the some of the packet commands that are sent to/from the USB device:

- Reset FIFO
- Enable/disable PC data transfers
- Read GPIF status register data
- Read GPIF transfer count
- SE4120L mode selection

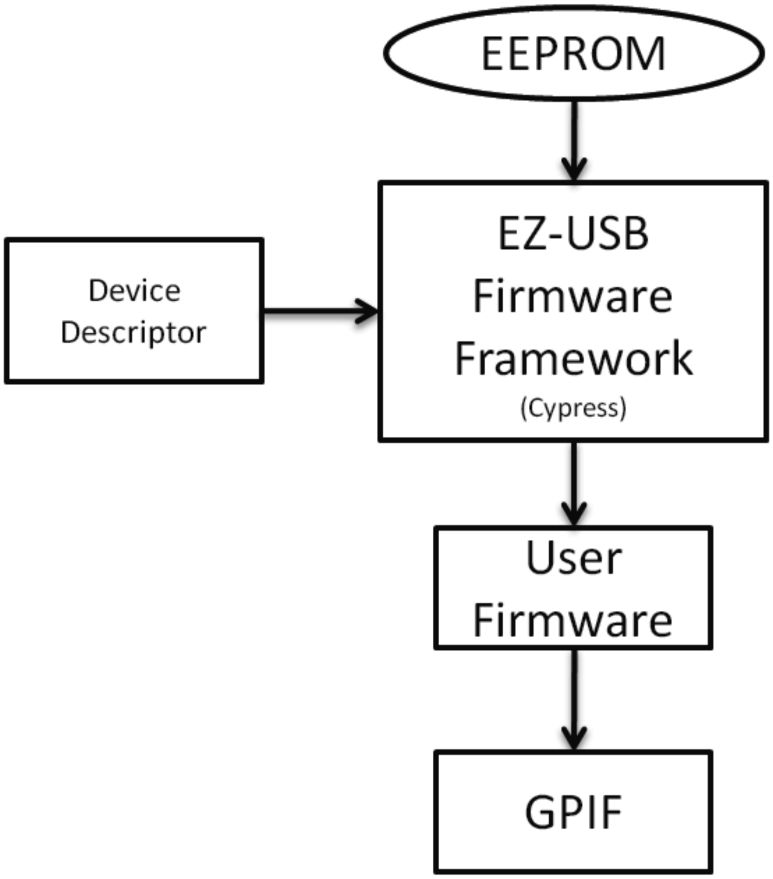
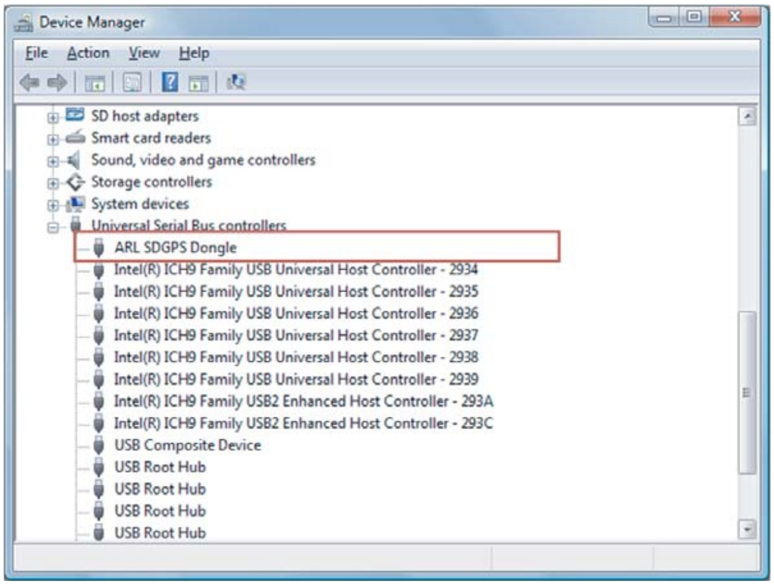


Figure 12. Device firmware.

5.2 USB Driver

The driver as in the device firmware was built around the CyUSB.sys driver, which is included in the EZ-USB framework. The driver provides a complete software development kit for using the underlying software and dynamic-linked library (DLL). To configure the driver for use with the device, a plain text file used by Microsoft Windows for installation of software and drivers, a .inf file, is defined. The only modification to the .inf is to identify the device via VID and PID with the one selected in section 5.1. This allows Windows to identify the device as a USB device with the associated driver software. See reference 16 for details on how to operate the device driver. Figure 13 is a screenshot of the installed device under Windows Vista (Note: Under Windows Vista, driver signature enforcement must be disabled to use this driver.)



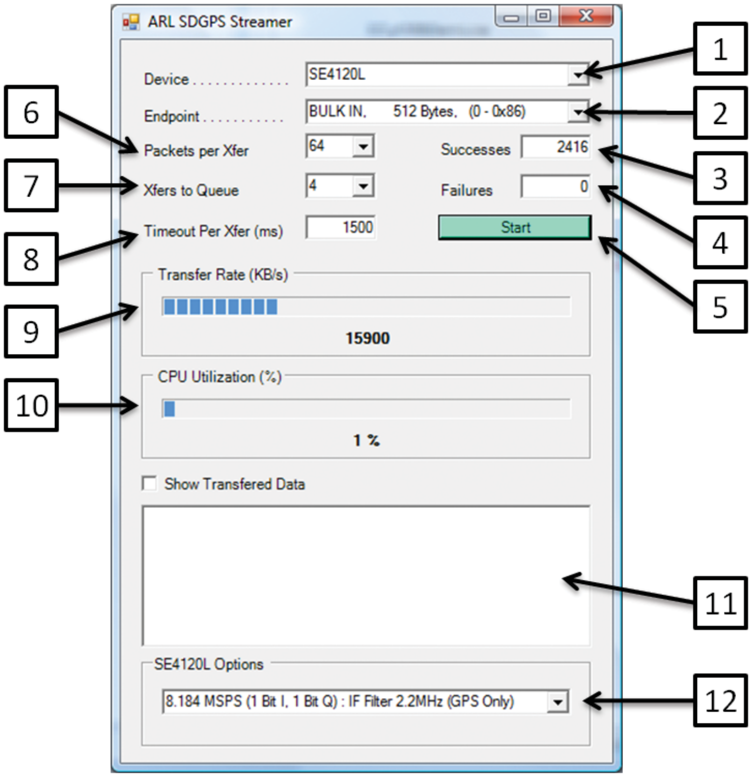
- SD host adapters
- Smart card readers
- Sound, video and game controllers
- Storage controllers
- System devices
- Universal Serial Bus controllers
  - ARL SDGPS Dongle
  - Intel(R) ICH9 Family USB Universal Host Controller - 2934
  - Intel(R) ICH9 Family USB Universal Host Controller - 2935
  - Intel(R) ICH9 Family USB Universal Host Controller - 2936
  - Intel(R) ICH9 Family USB Universal Host Controller - 2937
  - Intel(R) ICH9 Family USB Universal Host Controller - 2938
  - Intel(R) ICH9 Family USB Universal Host Controller - 2939
  - Intel(R) ICH9 Family USB2 Enhanced Host Controller - 293A
  - Intel(R) ICH9 Family USB2 Enhanced Host Controller - 293C
  - USB Composite Device
  - USB Root Hub
  - USB Root Hub
  - USB Root Hub
  - USB Root Hub

Figure 13. Device installation.

5.3 Capture Software

A user application software was developed to interface the Windows driver and to configure the device for operation. Figure 14 shows an overview of the software.

1. Device selection box, switches between SE4120L and MAX2769 Devices
2. Endpoint selection box, determines which endpoint to stream, (EP6 only)
3. Number of successful data transfers indicator
4. Number of failed data transfers indicator



Device SE4120L

Endpoint . . . . BULK IN, 512 Bytes, (0 - 0x86)

Packets per Xfer 64

Successes 2416

Xfers to Queue 4

Failures 0

Timeout Per Xfer (ms) 1500

Transfer Rate (KB/s) 15900

CPU Utilization (%) 1 %

□ Show Transfered Data

SE4120L Options 8.184 MSPS (1 Bit I, 1 Bit Q) : IF Filter 2.2MHz (GPS Only)

Figure 14. Capture software GUI.

5. Start/stop data streaming button
6. Number of packets per transfer
7. Number of transfers per queue
8. Packet transfer timeout
9. Current sustained data rate
10. CPU utilization
11. Displays transferred data and errors
12. Configuration box for SE4120L or MAX2769

The software interfaces with the EZ-USB framework using the C++ programming language. On startup, detection of the device is initiated and the Endpoint Combo box is loaded with available endpoints, in this case, only EP6. Data transfer is initiated on the device by first sending a Start command to the firmware to begin GPIF operation. When this occurs, a separate thread is executed to begin streaming the data to the hard drive from the driver software. The number of packets and number of loops initiated are controlled by the number of packets per transfer (6) and number of transfers per queue (7). These are essentially nested loops that control the amount of data within a large buffer. During this thread call, data are copied from the driver to a local buffer and then written to the hard drive in binary format. There are several events that may trigger a failure; (4) no. 4 in figure 14, a sync word at the boundary between buffers being inconsistent, a hard drive write error, and a timeout for the driver. Each of these results in the binary file being re-initialized. Figure 15 shows the flow of the data from the SE4120L configured for 1-bit IQ data. As shown, the data must go through several steps before being written to the hard drive. If there are excessive communication lags, the FIFO will overflow and trigger a failure in serializing the data stream. In an effort to catch some of these errors, the sync bit, bit 1, is compared at the first byte in the buffer, B_0 and the final byte in the buffer, B_{N-1}. If these bits match, then we have a loss of signal, as shown in table 4.

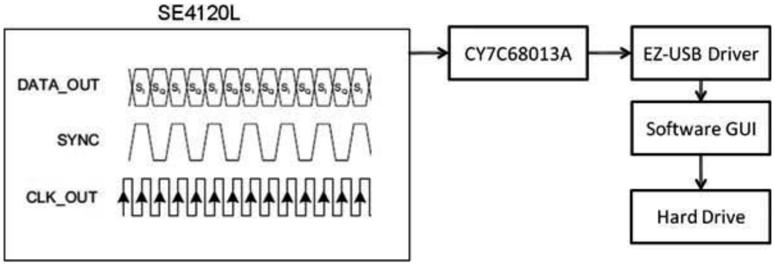


Figure 15. Data flow.

Table 4. Stream error identification.

Sample Number	Good Stream	Bad Stream
0	0000 001I	0000 001I
1	0000 000Q	0000 000Q
2	0000 001I	0000 001I
3	0000 000Q	0000 000Q
4	0000 001I	0000 001I
⋮	⋮	⋮
2^{N-1}	0000 000Q	0000 001I

5.4 Validation Software

The validation software used in this report is provided within the reference 1. This method was chosen to minimize the development time and for testing purposes. In the future, expansion upon this base software would inevitably include unique and improved software processing algorithms.

Figure 16 shows the flow of the GPS signal processing algorithm. The first step, Acquisition, takes the IQ data stream and determines which satellites are visible. This sets up the data structure that then defines the number of channels to process based on the number of satellites available. The information about the satellites to be tracked is passed to a tracking routine that provides the output of the discriminators and the correlation. All tracking is performed serially on these channels, using the scaler tracking loop method. After tracking the channels, the information is sent to a navigation routine, which calculates the pseudoranges and computes the position via the least squares method. Although these function are all rudimentary, they are used only as a validation of the system design.

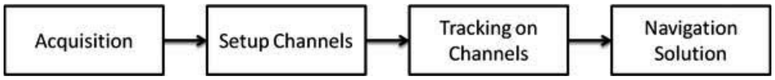


Figure 16. GPS software flow.

6. Experimental Results

Two experiments were conducted to validate the system, one under a simulated satellite constellation using a Spirent GNSS8000 simulator and one under open sky conditions. The two experiments were chosen to compare performance within the laboratory to outdoor real-world conditions.

The Spirent simulator was configured to run a simulation at latitude $32^o55'33.4255$” and longitude $-113^o43'50.255$” at a height of –121 m mean sea level on July 21, 2011 at 10 AM 19 min and 24 s. The latest Yuma almanac data files were used in the simulation with C/A code broadcast on L1 with no signal degradations or multi-path introduced. Since the RF frontend of the receiver is designed for an active antenna, a bias tee was used to block the DC to the simulator. Adjustments in the signal power were made to account for loss in the cabling and bias tee and lack of an eternal LNA. The signal levels were modeled using the $1/r^2$ propagation, a Klobuchar ionosphere propagation model, and Yuma files for the week *(18)*.

The hardware in section 3 was configured using a sampling rate of 8.1838 MHz with 1-bit IQ data. The signal was recorded for 60 s and using the software described in reference 1. Figure 17 shows the eye diagram of the IQ data from the receiver, observed from the peak correlator output, I_P and Q_P of one of the tracked satellites, PRN-17.

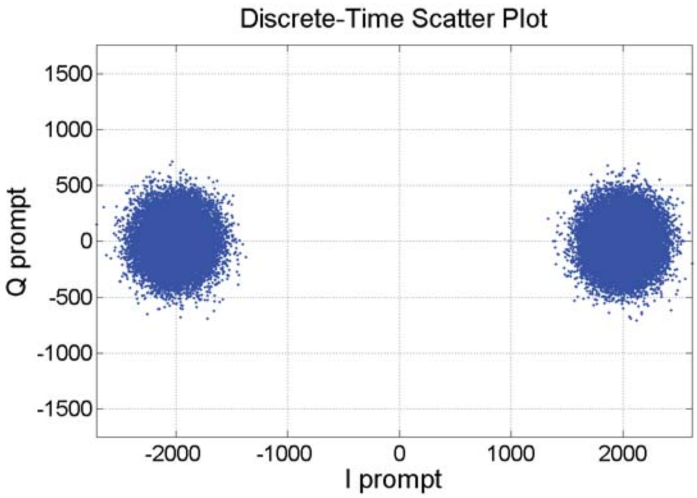


Figure 17. IQ eye diagram.

Due to the BPSK nature of the signal, we would expect to have this sort of distribution in the IQ plane. The mean separation of the points from the origin is approximately equal to the signal power and the variance on this distribution could be thought of as the noise. A good metric for receiver performance is the carrier-to-noise power ratio, $\frac{C}{N_0}$, calculated via equation 4:

$$\left(\frac{C}{N_0}\right)_{(dB-Hz)} = 10\log_{10}\left(\frac{E\left\{{I_P}^2\right\}N_{BW}}{E\left\{{Q_P}^2\right\}f_s\tau}\right) \tag{4}$$

where E is the expected value; I_P and Q_P is the output of the peak of the I and Q correlations, respectively, f_s is the sampling rate; N_{BW} is the bandwidth of the input filter; and τ is the integration and dump time constant. This calculation method outputs the values in dB-Hz, which provides a means for comparison to other receivers with difference sampling rates, noise bandwidths, etc. Figure 18 shows the $\frac{C}{N_0}$ for the simulator experiment.

In this simulation, five satellites were tracked with the average carrier to noise of 40.2 dB-Hz. Since we have more than four satellites in track, we can decode the navigation messages from the satellites and perform the navigation solution. The navigation message for PRN-17 is displayed in figure 19.

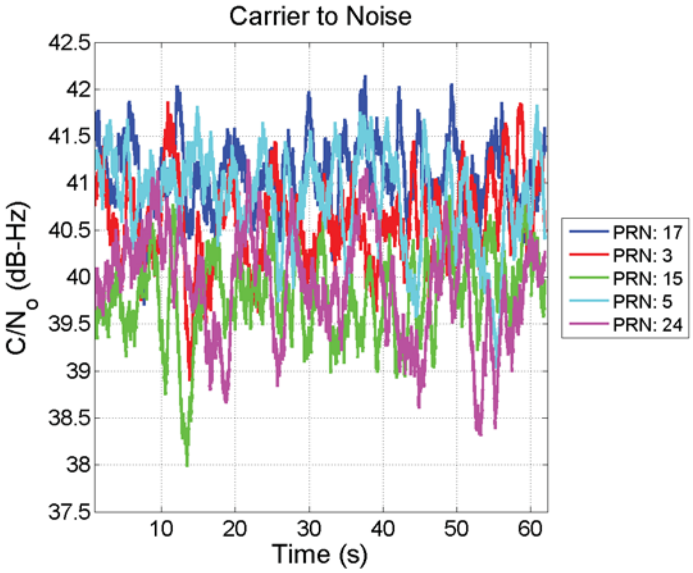


Figure 18. Carrier-to-noise power $\frac{C}{N_0}$.

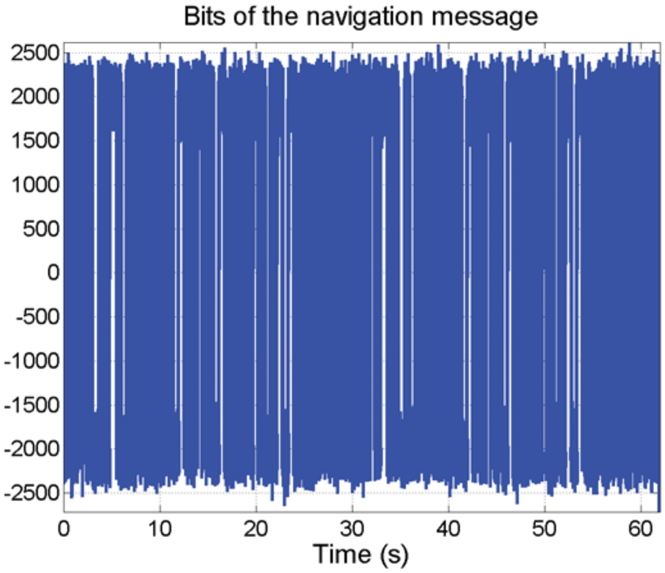


Figure 19. Navigation message, $d(t)$.

The message data are essentially the output of the I_P correlator. These data are therefore down-sampled to the navigation data rate of 50 bps and the information is extracted. Figure 20 shows one of the data sets available within the navigation message, the azimuth and elevation of the SVNs, known as a Skyplot.

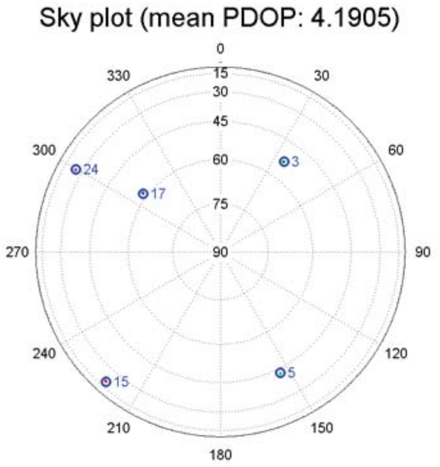


Figure 20. Spirent GNSS8000 simulation Skyplot.

Skyplot is an excellent tool for validatig the navigation message decoding process as it provides a visual reference to compare to known values of the satellites orbits. Figure 21 shows the results of the solution for the GPS receiver after the full processing of the raw IQ data stream.

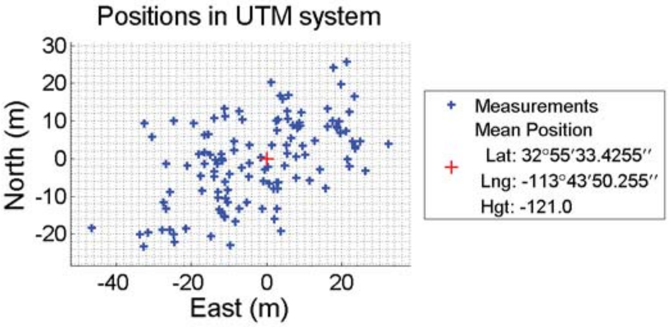


Figure 21. Spirent GNSS8000 simulation position results.

We can see that the position has a mean error of approximately 17 m in the $x - y$ plane. Although this position error is pretty significant, there is no correction for the ionosphere, troposphere delay, or compensation for the ephemeris; many other simplifications were made in the calculations. The results here are only to provide an indication that the data collection was sufficient for calculating a position.

The second experiment was conducted in the parking lot of Building 4600 at Aberdeen Proving Ground, MD. The data were collected on Friday, July 22, 2011, at approximately 9:00 AM local universal time (UTC). The receiver was connected to an Antenna Factor SH GPS antenna with total antenna gain of 28 dB and a <2-dB NF. In this circumstance, we were able to track eight satellites with the average $\frac{C}{N_0}$ of 39.04 dB, figure 22. Figure 23 shows the Skyplot of the results of the tracking of the SVN.

For this experiment, there was no surveyed antenna position for a measure of truth; therefore, all measurement errors are assumed about the mean measurement of the data collected. Figure 24 shows the position measurements relative to the mean of the measurements in the outdoor experiment.

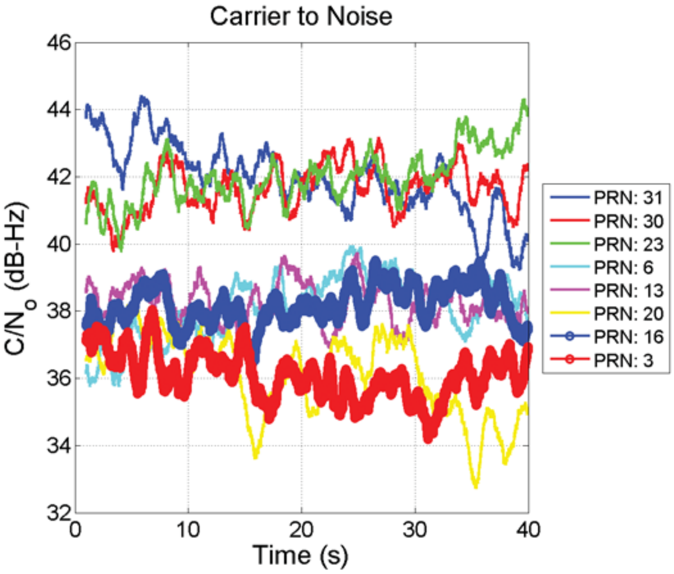


Figure 22. Field experiment $\frac{C}{N_0}$.

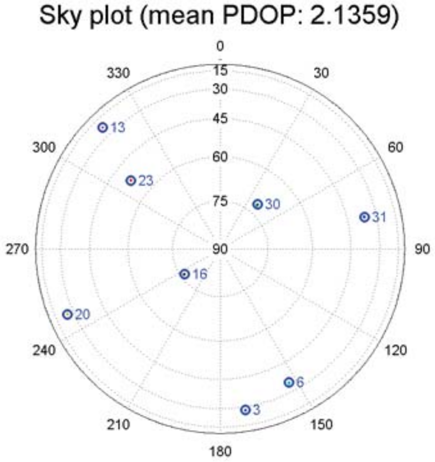


Figure 23. Field experiment Skyplot.

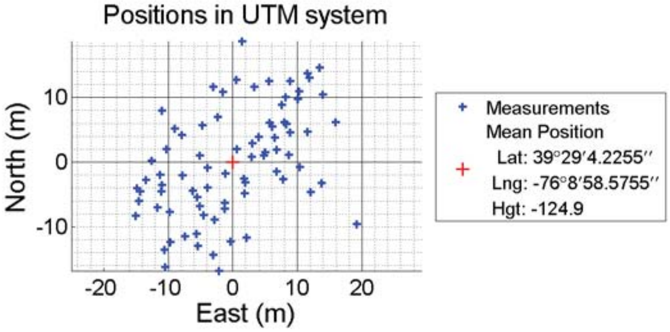


Figure 24. Field experiment position results.

7. Conclusion

The software for a GPS framework was presented and effectively demonstrated in the laboratory environment and in real-world outdoor conditions. Using a USB to acquire raw IQ measurements is shown to be an effective means of capturing GPS data to a hard drive for post-processing analysis on a computer. This device will provide developers with a measurement tool for capturing data for evaluation of signal processing techniques and tracking algorithms for more accurate GPS measurements. Results of experiments showed similar performance between outdoor conditions, using actual SVN and laboratory simulations. Later experiments would include tight integration of GPS measurements with inertial measurement unit (IMU) to improve tracking results.

Proceeding forward with this system, we will begin developing simulation models for GPS signals on projectiles. These simulation models would be validated using the SDR system to ensure simulations match reality. With proper simulation, we can begin developing algorithms tailored to the complex dynamics associated with spinning gun-launched projectiles. By merging ARL sensor models and flight dynamics within the PRESIMEN framework with the GPS simulation and data capturing technique outline in this report, complex algorithms can be rigorously evaluated. This capability will provide an avenue for developing advanced algorithms for gun-launched guided munitions, including deep coupling methods and anti-jam (AJ) techniques, ultimately increasing the Government's understanding of GPS systems within this environment.

References

[1] Borre, Kai; Akos, Dennis M.; Bertelsen, Nicolaj; Rinder, Peter, Jensen, S⊘ren Holdt. *A Software Defined GPS and Galileo Receiver: A Single Frequency Approach.* Birkhäuser, 2007.

[2] Analog Devices. *AD9235 Datasheet*, 2004.

[3] Feickert, Andrew. The Joint Tactical Radio System (JTRS) and the Army's Future Combat System (FCS): Issues for Congress. Technical report, National Defense Foreign Affairs, 2005.

[4] Friis, H. T. Noise Figures in Radio Recievers. *IEEE Proceedings of the IRE*, 1944.

[5] Gilmore, Rowan; Besser, Les. *Practical RF Circuit Design for Modern Wireless Systems*. *2*, Artech House, 2003.

[6] Gold, R. Optimal Binary Sequences for Spread Spectrum Multiplexing. *IEEE Transactions on Information Theory* **1967**, *13* (4). 619–621.

[7] Grewal, Mohinder S.; Weill, Lawrence R.; Andrews, Angus P. *Global Positioning Systems, Inertial Navigation, and Integration*. Wiley, 2 edition, 2007.

[8] Hoffman-Wellenhof, B.; Lichtenegger, H.; Collins, J. *GPS Theory and Practice*. Springer-Verlag Wien New York, 5 edition, 2001.

[9] Ilg, Mark; Fresconi, Frank; Maley, James; Alexander, Steven. GPS Roll Estimation for Gun-launched Spinning Projectiles: Hardware-in-the-loop Simulation and Experimental Validation. *Institute of Navigation, JNC Proceedings*, 2011.

[10] Kaplan, Elliot D.; Hegarty, Christopher J. *Understanding GPS Principles and Applications*.Artech House, Inc., 2 edition, 2006.

[11] Katulka, G.; Hall, Rex. Electromagnetic Analysis of 60mm Mortars Modified with Guidance, Navigation and Control Components. *IEEE International Symposium on Antennas and Propagation Proceedings* July, 2010.

[12] GPSW LAAFB. Navstar GPS Space Segment / Navigation User iInterfaces: Icd-gps-200e. Technical report, GLOBAL POSITIONING SYSTEM WING (GPSW) SYSTEMS ENGINEERING & INTEGRATION, 1993.

[13] Cypress Semiconductor. Cy7c68013 datasheet. *www.cypress.com*, 2002.

[14] Cypress Semiconductor. Ez-usb technical reference manual. *www.cypress.com*, 2002.

[15] Cypress Semiconductor. Introduction to the ez-usb fx2 gpif engine. *www.cypress.com*, 2002.

[16] Cypress Semiconductor. Cypress cyusb.sys programmer's refernce. *www.cypress.com*, 2010.

[17] SiGe Semiconductor. Se4120l datasheet. *www.sige.com*, 2009.

[18] Spirent. *GSS8000 GNSS Constellation Simulator Manual*, 2010.

NO. OF
COPIES ORGANIZATION

1 ADMNSTR
ELEC DEFNS TECHL INFO CTR
ATTN DTIC OCP
8725 JOHN J KINGMAN RD STE 0944
FT BELVOIR VA 22060-6218

1 US ARMY RSRCH DEV AND ENGRG
CMND
ARMAMENT RSRCH DEV & ENGRG
CTR ARMAMENT ENGRG &
TECHNLGY CTR
ATTN AMSRD AAR AEF T
J MATTS
BLDG 305
ABERDEEN PROVING GROUND MD
21005-5001

1 CDR US ARMY TACOM ARDEC
ATTN RDAR MEF E T RECCHIA
BLDG 94
PICATINNY ARSENAL NJ 0706-5000

1 CDR US ARMY TACOM ARDEC
ATTN SFAE AMO CAS P MANZ
BLDG 172
PICATINNY ARSENAL NJ 0706-5000

1 CDR US ARMY TACOM ARDEC
ATTN R B BUCKELEW
BLDG 5400
REDSTONE ARSENAL AL 35898-5000

1 US ARDEC
ATTN RDAR MEF M HOHIL
BLDG 407
PICATINNY ARSENAL NJ 07806-5000

1 US ARMY ARDEC
ATTN RDAR MEF
WILLIAM R SMITH
ATTN RDAR MEM C P MAGNOTTI
BLDG 94
PICATINNY ARSENAL NJ 07806-5000

NO. OF
COPIES ORGANIZATION

1 US ARMY INFO SYS ENGRG CMND
ATTN AMSEL IE TD A RIVERA
FT HUACHUCA AZ 85613-5300

1 COMMANDER
US ARMY RDECOM
ATTN AMSRD AMR S CORNELIUS
5400 FOWLER RD
REDSTONE ARSENAL AL 35898-5000

1 US ARMY RDECOM/ARDEC
ATTN J PELINO
BLDG 93 3RD FLOOR
PICATINNY ARSENAL NJ 07806-5000

1 US GOVERNMENT PRINT OFF
DEPOSITORY RECEIVING SECTION
ATTN MAIL STOP IDAD J TATE
732 NORTH CAPITOL ST NW
WASHINGTON DC 20402

1 US ARMY RSRCH LAB
ATTN RDRL WML F M ILG
BLDG 4600
ABERDEEN PROVING GROUND MD
21005

3 US ARMY RSRCH LAB
ATTN IMNE ALC HRR MAIL &
RECORDS MGMT
ATTN RDRL CIO LL TECHL LIB
ATTN RDRL CIO LT TECHL PUB
ADELPHI MD 20783-1197

TOTAL: 15 (14 HCS, 1 ELEC)

INTENTIONALLY LEFT BLANK.